

Lego Datenbank

Lerne auf spielerische Art, wie Daten modelliert und in einer Datenbank gespeichert sowie abgerufen werden können.

Die Lego Datenbank soll Dir zeigen, warum die Datenmodellierung wichtig ist und welche Bausteine du dazu benötigst. Dazu gehören die Entitäten (Dinge), also zum Beispiel der Legostein, dessen Eigenschaften (wie z. B. Farbe oder Form) sowie die Verbindungen zu anderen Legosteinen oder zu den Figuren. Die Datenmodellierung beschreibt also „WAS“ Du in deiner Lego Datenbank (Legosammlung) speichern bzw. aufbewahren möchtest und wie die Dinge zueinander in Verbindung stehen.

Dieser Workshop deckt diesen Bereich der Cloud Computing Lab Matrix ab:

Datenbanken	Strukturierung von Information (Entity-Relationship Modell)	X	X	X
	Datenbankabfragen (Structured Query Language)		X	X
	Fortführende Konzepte (BigData und noSQL)			X

Besuche uns auf unserer Homepage

<https://cola.fh-joanneum.at/>



Viel Spaß wünscht das CoLa Team

Mathias Knoll, Harald Schwab, Stefanie Wieser, Priska Steininger, Michael
Brickmann, Elmar Krainz, Petra Kletzenbauer, Sabine Proßnegg, Rappl
Stefan

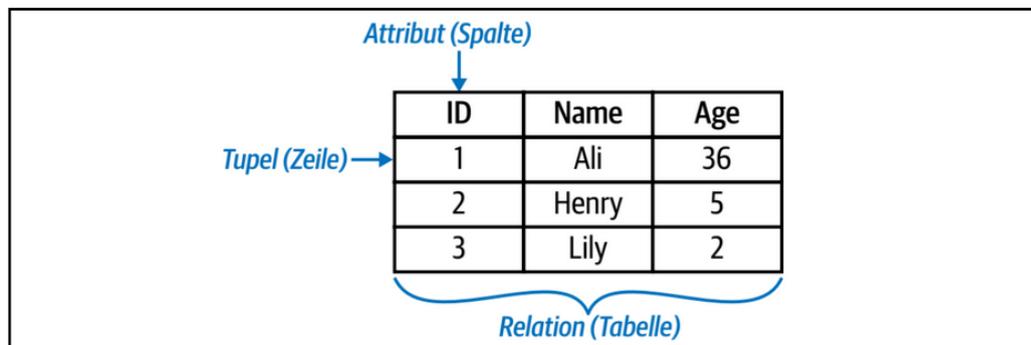
Inhalt

Basisspiel	4
Dinge (Entitäten).....	5
Eigenschaften von Dingen	5
Verbindungen zwischen Dingen	Fehler! Textmarke nicht definiert.
Aufgaben.....	7
Erweiterung 1.....	10
Tabellen	10
Datentypen	11
Lego-Id als Primärschlüssel	11
Aufgaben.....	13
Erweiterung 2.....	14
Datenbank-Abfragen	14
Aufgaben.....	17

Basisspiel

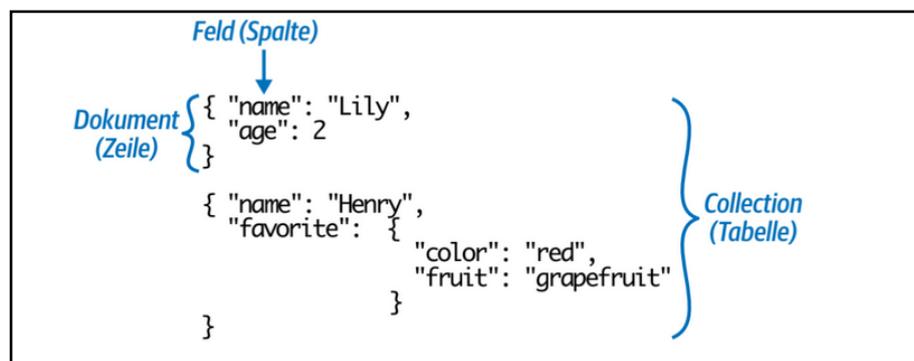
Zunächst soll der Begriff „Datenbank“ spezifiziert werden. Eine Datenbank ist ein Ort, wo Sie auf organisierte Weise Daten ablegen bzw. speichern können. Grundsätzlich gibt es viele Datenbanken, die Sie hierfür verwenden können. Die zwei wesentlichen Datenbank-Kategorien sind aber SQL und NoSQL (vgl. Zhao 2022).

SQL bedeutet Structured Query Language, also strukturierte Abfragesprache. SQL-Datenbanken werden auch als relationale Datenbanken bezeichnet, da sie aus Relationen/Tabellen bestehen. Dies bedeutet, dass viele Tabellen, die miteinander verbunden sind, eine Datenbank bilden. An dieser Stelle ist es wichtig zu erwähnen, dass SQL-Datenbanken stets ein vordefiniertes Schema erfordern. Bevor Sie also eine Tabelle anlegen, müssen Sie die Struktur entsprechend festlegen wie etwa Spalten der Tabelle und die jeweiligen Datentypen wie z. B. Ganzzahl oder Zeichenketten etc. (vgl. ebd.).



Quelle: Zhao 2022

Lassen sich Daten nicht mehr in einer strukturierten Weise organisieren, benötigen Sie eine NoSQL-Datenbank. Dies bedeutet Not only SQL, also nicht nur SQL. Eines der wesentlichen Merkmale von NoSQL ist, dass sie dynamische Schemata besitzen, was bedeutet, dass ihre Schemata nicht im Vorhinein festgelegt werden müssen. Die Daten liegen also nicht mehr in einer strukturierten Tabelle vor. Die nachfolgende Abbildung zeigt, wie die Daten in einer NoSQL-Datenbank wie MongoDB gespeichert werden (vgl. ebd.).



Quelle: Zhao 2022

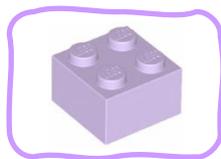
Eine Datenbank ist der Ort, indem Du Deine Dinge wie z. B. Legosteine oder Figuren aufbewahren kannst. Diese Datenbank oder "Sammlung" beinhaltet verschiedene Dinge wie Legosteine, Figuren usw., die wiederum über verschiedene Eigenschaften wie Farbe=**Rot**, Form=**Würfel** verfügen können. Was du in deiner Lego Datenbank aufbewahrst und wie die Bausteine zueinander in Verbindung stehen, wird in Form des Entity-Relationship-Modells beschrieben.

Dinge (Entitäten)

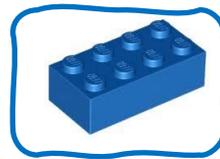
Entitäten sind Dinge, über die wir Informationen sammeln möchten. In unserem Spiel sind dies die Legosteine oder auch Figuren. Jeder Legostein – zum Beispiel der grüne Stein mit der Form 2x3 gebogen oder die Figur Flamingo – stellen für sich eine Entität/Ding dar.



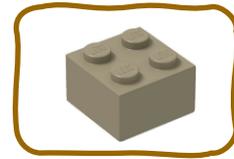
Lego1



Lego2



Lego3



Lego4

Flamingo



Eigenschaften von Dingen

Jedes Ding kann über verschiedene Eigenschaften oder Merkmale verfügen. Zum Beispiel könnte unser Legostein die Eigenschaften Farbe und Form haben.



LEGO	
NUMMER	1
FARBE	rot
FORM	Stein 2x4 (Block)

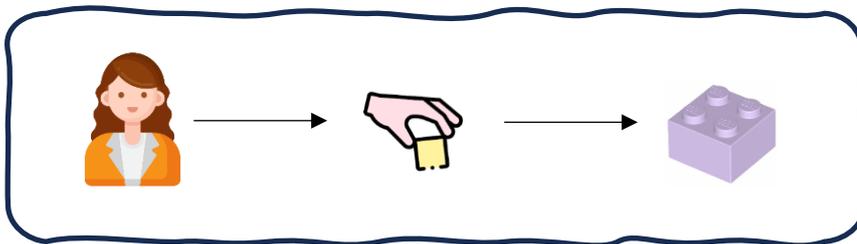
Verbindungen zwischen Dingen

Neben der Entität (Ding) also dem Legostein selbst spielt auch die Verbindung zu anderen Legosteine oder Figuren eine große Rolle. Denn ein Legostein steht mit einer Figur in Verbindung oder anders ausgedrückt eine Figur wie z. B. der Flamingo benötigt einen oder viele Legosteine, um seine Gestalt anzunehmen.

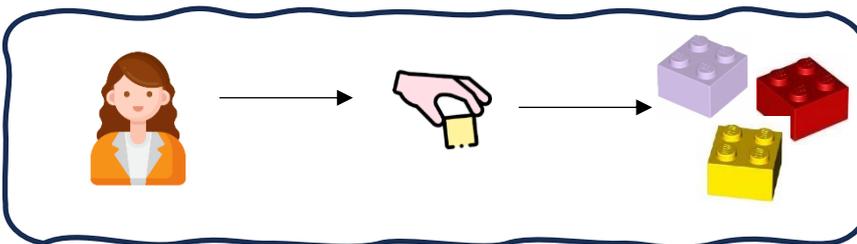
Im Allgemeinen können zwischen drei wesentlichen Verbindungen unterschieden werden. Dies wird anhand des Spiels skizziert:

- **1:1 Verbindung:** Dies bedeutet beispielsweise, dass **ein Schüler einen Legostein aus der Sammlung entnimmt** oder daher eine Figur lediglich einen Legostein benötigen würde, um seine Gestalt anzunehmen.
- **1:n Verbindung (1: viele):** Dies trifft schon eher auf unser Spiel zu, denn das heißt, dass **ein Schüler viele Legosteine aus der Legosteine-Datenbank entnimmt**, um sein Muster auf der Karte zusammenzubauen. Das bedeutet daher auch, dass **eine Figur viele Legosteine benötigt**, um seine Gestalt anzunehmen.
- **m:n Verbindung (viele:viele):** Auch das ist eine Verbindung, die in unserem Spiel zutrifft, denn **viele Schüler aus der Klasse müssen viele Legosteine aus der Lego-Datenbank entnehmen**, um ihre Figuren zusammenbauen zu können (=da **viele Figuren auch viele Legosteine benötigen**, um ihre Gestalten anzunehmen).

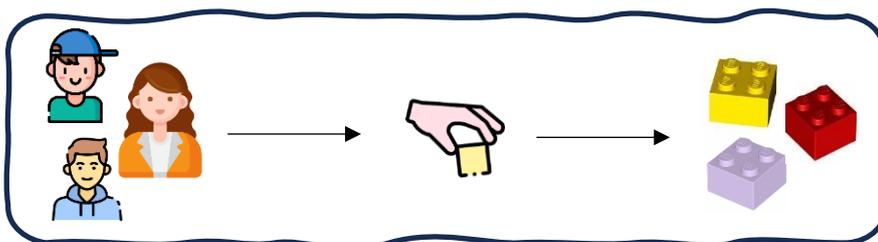
1:1 Verbindung



1: viele Verbindung



viele:viele Verbindung



Datenmodellierung, ER-Modell und Relationen

Die Datenmodellierung ist ein wesentlicher Faktor bei der Entwicklung eines Informatiksystems. Es muss ein Datenbank-Schema gefunden werden, sodass alle erforderlichen Daten im Datenbank-Schema abgespeichert werden können, die Konsistenz der Daten sichergestellt wird sowie schließlich effizient auf die Daten zugegriffen werden kann (vgl. Studer 2019).

Zur Beschreibung der Daten umfasst das ER-Modells folgende Elemente – Entitäten bzw. Entitätstypen, Beziehungen und Attribute. Entitätstypen stellen Aspekte der realen Welt auf einem abstrakten Niveau dar. Beispielsweise könnte ein konkreter Entitätstyp ein bestimmter Kunde oder auch ein bestimmter Artikel sein. Beziehungstypen spezifizieren den Zusammenhang zwischen den Entitäten. Das bedeutet zum Beispiel bei einem Onlineshop, dass Kund:innen auch verschiedene Artikel bestellen können. Attribute beschreiben Entitäten oder Beziehungen genauer. Beispielsweise können Kunden mitunter über die Attribute „Name“, „Anschrift“, „Kundenklasse“, etc. – verfügen und dadurch näher beschrieben werden (vgl. Gadatsch 2017).

Relationen und ihre Tabellen lassen sich nicht isoliert betrachten. Zu Illustration lässt sich als Beispiel eine Bestellung anführen, die diese mit ihren Bestellpositionen und den Daten des Rechnungsempfängers in einer Beziehung stehen. Somit werden für eine Bestellung mindestens drei Relationen benötigt: eine für die Bestellung, eine für die Bestellpositionen und eine weitere für die Rechnungsempfänger (vgl. Ernst et al. 2016).

Abhängig von den Kardinalitäten können die Relationen wie folgt unterschieden werden:

1 zu 1 Beziehung	Zu einem Tupel gehört genau ein anderes Tupel. Beispiel: Person ↔ Kopf. Oder zu einem Tupel gehört ein oder kein anderes Tupel (Konditional). Beispiel: Person ↔ Reisepass
1 zu N Beziehung	Zu einem Tupel gehört mindestens ein (≥ 1) anderes Tupel. Beispiel: Mitarbeiter → Adresse. Oder zu einem Tupel gehören beliebig viele (≥ 0) andere Tupel. Beispiel: Ehepaar → Kinder
M zu N Beziehung	Zu einem Tupel gehören beliebig viele (≥ 0) andere Tupel umgekehrt gilt dasselbe. Beispiel: Auto — Fahrer

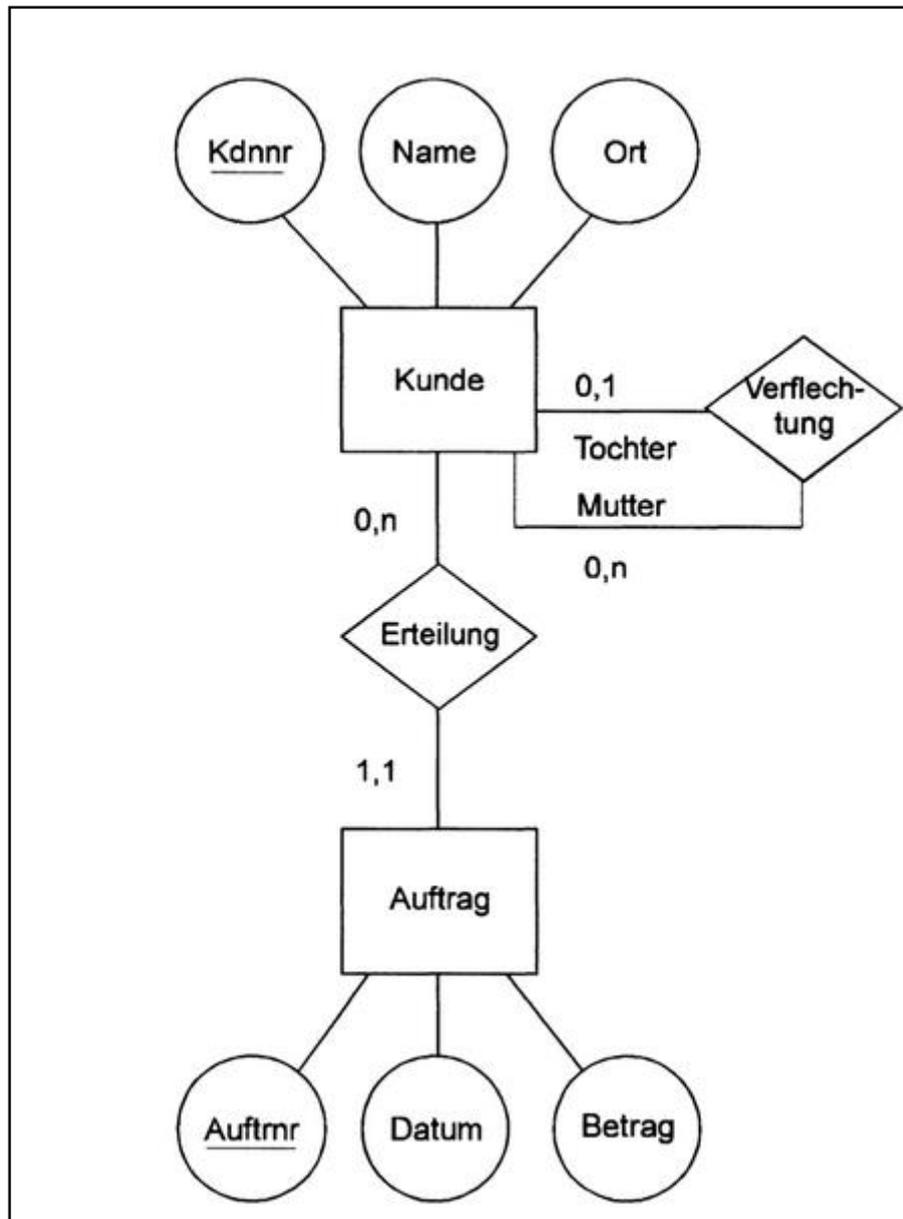
Quelle: Ernst et al. 2016

Zusätzliche Informationen erhalten Sie unter den Links:

[https://de.wikipedia.org/wiki/Relation_\(Datenbank\)](https://de.wikipedia.org/wiki/Relation_(Datenbank))

https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model

Nach Stickel (vgl. 2013) soll nachfolgendes ER-Diagramm beispielhaft den Ausschnitt aus dem Schema eines Betriebs darstellen, wie es auch in der Praxis anzutreffen ist.



Quelle: Stickel 2013

Die Objekte (Entitäten) werden im Diagramm als Kästchen, die verschiedenen Beziehungsarten als Rauten und die Attribute als Ovale dargestellt. Dies bedeutet, dass Sie sämtliche Aspekte und Eigenschaften angeben, die Sie in der Datenbank speichern möchten. Ein unterstrichenes Attribut wie Auftrnr repräsentiert einen Schlüssel wie den Primärschlüssel, der im nächsten Abschnitt genauer erläutert wird. Die Einträge wie *1,1* stehen für die jeweilige Kardinalität der Beziehungsart, die Sie bereits kennen gelernt haben. Beispielsweise würde die Kardinalität *0,n* bei der Entität Kunde und der Beziehung Erteilung in diesem Fall bedeuten, dass jeder Kunde mindestens null oder mehrere Aufträge erteilt hat (vgl. ebd.).

Aufgaben

1. Nimm einen Legostein aus der Legosteine-Sammlung (Datenbank).
2. Beschreibe diesen Legostein.
3. Zieh eine Karte.
4. Baue das Muster auf der Karte.

Erweiterung 1

Tabellen

In einer relationalen Datenbank werden die Daten in Form von Tabellen angeordnet. Dies bedeutet, dass unsere Lego-Datenbank aus einer Reihe von Tabellen besteht. Wir haben also eine Tabelle für die Legosteine, eine weitere für die Figuren und eine Verknüpfung der beiden Tabellen. Diese Tabelle sehen folgendermaßen aus:

Auszug aus der Tabelle – Legosteine

(vollständige Tabellen im Anhang)

Lego-Id	Farbe	Form	Anzahl
1	rot	Stein 2x2 (dünn)	2
2	dunkelgrün	Stein 2x2 (dünn)	1
3	hellblau	Stein 2x2 (dünn)	3
4	hellgrün	Stein 2x2 (dünn)	10
5	blau	Stein 2x2 (dünn)	4
6	weiß	Stein 2x2 (dünn)	3
...

Auszug aus der Tabelle – Figuren

Figur-Id	Name	Anzahl Legosteine
1	Flamingo	8
2	Wal	10
3	Elefant	14
4	Eichhörnchen	9
5	Schildkröte	8
6	Krokodil	11
7	Ente	8
8	Kamel	12
..

Auszug aus der Verknüpfung der Tabellen

Figur-Id	Name	Lego-Id	Anzahl pro Lego
1	Flamingo	32	1
1	Flamingo	9	1
1	Flamingo	44	1
1	Flamingo	17	1
1	Flamingo	40	1
1	Flamingo	25	1
1	Flamingo	36	1
1	Flamingo	43	1
...

Anhand dieser Tabellen musst du die richtigen Steine für dein Muster in deine eigene Tabelle eintragen.

Datentypen

Die Daten in einer Datenbank können unterschiedliche Werte beinhalten. Beispielsweise kann die Lego-Id eine Zahl oder der Name eine Zeichenkette sein. Dementsprechend sind in den Tabellen neben den Eigenschaften einer Entität (z. B. Legosteine) auch der jeweilige Datentyp festzulegen. In der Lego-Datenbank werden vor allem die folgenden Datentypen verwendet:

Datentyp	Beschreibung
INT	Steht für eine positive oder negative ganze Zahl
VARCHAR	Bedeutet ein Text variabler Länge
CHAR	Bedeutet eine bestimmte Textlänge
<i>Weitere:</i>	
FLOAT, DOUBLE	Wird für Gleitkommazahlen verwendet
DATE	Wird für Datumsangabe genutzt

Vgl. <https://www.peterkropff.de>, <https://www.lucidchart.com>

Zusätzliche Informationen finden Sie unter diesem Link:

https://de.wikibooks.org/wiki/Einf%C3%BChrung_in_SQL:_Datentypen

Lego-Id als Primärschlüssel

Wie ihr sehen könnt, enthält die Legosteine-Tabelle viele Zeilen. Es ist jedoch wichtig, die Datensätze (Zeilen) in einer Tabelle eindeutig zu kennzeichnen, um die Legosteine einfacher den Figuren zuzuordnen bzw. besser damit arbeiten zu können.





Der Primärschlüssel der Legosteine-Tabelle bezieht sich auf die Eigenschaft Lego-Id (Identifikationsnummer). Dieser dient also dazu, alle Legosteine vollständig voneinander zu unterscheiden.

Img src: [flaticon](#)

Zum Beispiel ist die Lego-Id der Primärschlüssel für jeden Legostein in der Legosteine-Tabelle und die Figur-Id der Primärschlüssel der Figuren-Tabelle.

Lego-Id 	Farbe	Form	Anzahl
1	rot	Stein 2x2 (dünn)	2
2	dunkelgrün	Stein 2x2 (dünn)	1
3	hellblau	Stein 2x2 (dünn)	3

Figur-Id 	Name	Anzahl Legosteine
1	Flamingo	8
2	Wal	10
3	Elefant	14

Mit diesen Schlüsseln können die Legosteine und die Figuren in der Datenbank eindeutig identifiziert bzw. erkannt werden.

Die Festlegung des Primär- und Fremdschlüssels sind die wichtigsten Integritätsbestimmungen in SQL (vgl. www.ibm.com 2021, <https://databasecamp.de> 2023):

- Der Primärschlüssel untersagt die Eingabe in einer oder mehreren Spalten innerhalb einer Tabelle doppelte Werte. Die Definition von Primärschlüssel gewährleistet, dass eine Spalte oder eine Kombination aus mehreren Spalten einen eindeutigen Wert für einen Datensatz beinhalten.
- Der Fremdschlüssel ist eine logische Regel über Werte in einer oder mehreren Spalten in einer Tabelle. Ein Fremdschlüssel ist ein Attribut, das in einer anderen Tabelle der Primärschlüssel ist wie z. B. im nachfolgenden Beispiel die AbteilungsNr. in der Tabelle Mitarbeiter. Hierdurch werden Verbindungen/Verknüpfungen zwischen Tabellen gewährleistet.

Genauere Informationen können Sie unter diesem Link nachlesen:

<https://databasecamp.de/daten/primaerschluessel>

Nachfolgendes praxisnahes Beispiel zeigt Ihnen Tabellen die Werte enthalten, um Mitarbeiter und Abteilungen in einer Datenbank zu speichern:

PersonalNr.	Name	Gehalt	AbteilungsNr.
101	Meissner	75 000	1
102	Lehman	98 000	1
103	Kunze	75 000	2

AbteilungsNr.	Name	Standort
1	Einkauf	München
2	DV-Org.	Augsburg
3	Produktion	Straubing
4	Entwicklung	Augsburg
5	Verwaltung	München

Verknüpfung der Tabelle mit Mitarbeiter mit Abteilungen

PersonalNr.	Name	Gehalt	AbteilungsNr.	AName	Standort
101	Meissner	75 000	1	Einkauf	München
102	Lehman	98 000	1	Einkauf	München
103	Kunze	75 000	2	DV-Org.	Augsburg

Quelle: In Anlehnung an Ernst et al. 2016

Bei diesem Beispiel erfolgt eine Verknüpfung mit dem Attribut *AbteilungsNr.*, welches in beiden Tabellen als Attribut vorhanden ist. In der Tabelle Abteilungen ist dies der Primärschlüssel und in der Tabelle Mitarbeiter der Fremdschlüssel. Dies führt dazu, dass die beiden Tabellen verknüpft werden können. Wie dies genau in der Erstellung der Tabelle deklariert wird, wird in Abschnitt 3 dargestellt.

Aufgaben

1. Ziehe eine Karte.
2. Trage in deiner Tabelle den Primärschlüssel und die Anzahl der jeweiligen Legosteine ein, die du für dein Muster benötigst.
3. Baue deine Figur.

Erweiterung 2

Datenbank-Abfragen

Um mit der Lego-Datenbank kommunizieren zu können, müsst ihr eine spezielle Abfragesprache „Structured Query Language (SQL)“ anwenden.

Eine SQL-Abfrage ist eine Anweisung, um Daten aus der Lego-Datenbank abzurufen oder zu bearbeiten. Mit diesen Abfragen kannst du beispielsweise auf die Tabelle Legosteine oder Figuren zugreifen, Legosteine filtern oder auch sortieren.

Nachfolgende Tabelle gibt dir einen Überblick über die wichtigsten SQL-Abfragen:

Abfrage	Beschreibung
SELECT (AUSWÄHLEN)	Mit SELECT kannst du die Daten aus einer oder mehreren Tabellen auswählen.
INSERT (EINFÜGEN)	Mit INSERT kann du neue Daten in eine Tabelle einfügen.
UPDATE (ÄNDERN)	Mit UPDATE kann du vorhandene Datensätze in einer Tabelle aktualisieren.
DELETE (LÖSCHEN)	Mit DELETE kann du Datensätze aus einer Tabelle löschen.
CREATE (ERZEUGEN)	Mit CREATE kannst du eine neue Tabelle oder Datenbank erstellen.
WHERE (FILTERUNG)	Mit WHERE kannst du die Datensätze auf bestimmte Bedingungen filtern.
JOIN (VERKNÜPFUNG)	Mit JOIN können Daten aus zwei oder mehreren Tabellen basierend auf einer gemeinsamen Spalte kombiniert werden.

vgl. <https://datascientest.com>

In diesem Spiel benötigst du insbesondere die SQL-Abfrage „SELECT“, da du die richtigen Legosteine für dein Muster **auswählen** musst.

Die Abfrage sieht folgendermaßen aus:

```
SELECT (AUSWAHL) Spaltenname FROM (VON) TABELLE;
```



Wenn du beispielsweise alle Legosteine aus der Legosteine-Tabelle auswählen möchtest, kannst du es so formulieren:

```
AUSWAHL * VON LEGOSTEINE;  
(SELECT * FROM LEGOSTEINE);
```

Das * bedeutet ALLE. Damit kannst du alle Legosteine aus dieser Tabelle auswählen.

Du benötigst aber NICHT alle Legosteine, sondern nur bestimmte, um dein Muster zusammenzubauen. Dies bedeutet, dass du einen Legostein nach einer bestimmten Bedingung auswählen musst. Das kannst du mit einer WHERE-Bedingung durchführen.

Zum Beispiel benötigst du für den Flamingo den Legostein mit der Lego-Id 1:

```
AUSWAHL Farbe, Form VON Legosteine WHERE (WO) Lego-Id=1
```

Mit dieser Abfrage erhältst du nur den Legostein mit der Id 1 und den Eigenschaften Farbe und Form.

Grundsätzlich lassen sich SQL-Anweisungen zwei wesentlichen Gruppen zuordnen – der DDL (Data Definition Language) und der DML (Data Manipulation Language). SQL umfasst Anweisungen, mit denen ein Datenbankschema wie etwa Tabellen etc. angelegt werden kann. Diese Anweisungen gehören zur DDL. Das Anlegen einer Tabelle wird beispielweise mit **CREATE TABLE** durchgeführt, dies nachfolgend beispielhaft dargestellt wird (vgl. Ernst et al. 2016). Dies bezieht sich wiederum auf die Tabelle Mitarbeiter des vorherigen Abschnitts.

```
CREATE TABLE Mitarbeiter (  
    PersonalNr.      int primary key,  
    Name             varchar(255),  
    Gehalt           int,  
    AbteilungsNr.int,  
    foreign key (AbteilungsNr.) references Abteilungen (AbteilungsNr)  
);
```

Zunächst muss die Tabelle mit *Mitarbeiter* benannt werden. Ebenso ist es erforderlich, zu jeder Spalte ein *Name* und ein *Datentyp* (siehe vorherigen Abschnitt) anzugeben. Eine Zeichenkette wird als *varchar* bezeichnet und darf in diesem Fall maximal 255 Zeichen lang sein. Darüber hinaus sind die Integritätsbedingungen (siehe vorherigen Abschnitt) zu definieren. Dazu gehören die *PersonalNr.* als Primärschlüssel und ferner die *AbteilungsNr.* als Fremdschlüssel (vgl. Ernst et al. 2016).

Zusätzliche Informationen finden Sie unter diesem Link:

<http://www.sql-lernen.de/create-table.php>



Bei der Abfrage einer Datenbank werden Daten meist aus einen oder mehreren Tabellen abgerufen. Hierfür benötigen Sie die SELECT-Klausel, welche Spalten angibt, die Anweisungen für Sie wiedergeben (vgl. Zhao 2022). Beispielsweise:

```
SELECT PersonalNr., Name FROM Mitarbeiter;
```

Ergebnis der Abfrage:

PersonalNr.	Name
101	Meissner
102	Lehman
103	Kunze

Mit folgender Abfrage können Sie alle Spalten aus einer Tabelle auswählen:

```
SELECT * FROM Mitarbeiter;
```

Ergebnis der Abfrage:

PersonalNr.	Name	Gehalt	AbteilungsNr.
101	Meissner	75 000	1
102	Lehman	98 000	1
103	Kunze	75 000	2

Um Daten in eine Tabelle einzufügen, zu ändern oder zu löschen ermöglicht SQL die Ausführung von drei Anweisungen – INSERT, UPDATE und DELETE. Diese Anweisungen gehören zur DML innerhalb von SQL (vgl. Ernst et al. 2016). Nennen wir das Beispiel von der Tabelle Mitarbeiter her und versuchen einen neuen Mitarbeiter einzufügen:

```
INSERT INTO Mitarbeiter VALUES (104, ‚Meier‘, 45 000, 4);
```

Bereits vorhandene Datensätze können mit UPDATE verändert werden. Dabei ist die WHERE-Bedingung wesentlich, da sie die zu verändernden Zeilen identifiziert (vgl. ebd.).

```
UPDATE Mitarbeiter SET Gehalt=95 000 WHERE Name=‘Lehman‘;
```

Ebenso können Daten wieder gelöscht werden. Auch hier ist die WHERE-Bedingung erforderlich, um genau jene Daten zu löschen, auf diese die Bedingung zutrifft.

```
DELETE FROM Mitarbeiter WHERE Name=‘Meier‘;
```

Weitere Informationen finden Sie unter:

<https://wiki.selfhtml.org/wiki/Datenbank/SQL-Grundlagen>



Aufgaben

1. Ziehe eine Karte.
2. Trage in der Tabelle den Schlüssel, die Anzahl der Steine, sowie die Abfrage an die Datenbank ein.
3. Baue deine Figur.



Referenzen

Datenbanken strukturieren. Abgerufen am 11. März, 2024, <https://www.lucidchart.com/pages/de/datenbankdesign-struktur>

Ernst, H., Schmidt, J., Beneken, G. (2016). Grundkurs Informatik: Grundlagen und Konzepte für die erfolgreiche IT-Praxis - Eine umfassende, praxisorientierte Einführung. Germany: Springer Fachmedien Wiesbaden.

Gadatsch, A. (2017). Datenmodellierung für Einsteiger: Einführung in die Entity-Relationship-Modellierung und das Relationenmodell. Germany: Springer Fachmedien Wiesbaden.

MySQL - SQL - Grundlagen- Datentypen. Abgerufen am 11. März 2024, <https://www.peterkropff.de/site/mysql/typen.htm>

Primärschlüssel: Definition, Vorteile und Besonderheiten (2023). Abgerufen am 11. März, 2024, <https://datascientest.com/de/primaerschluessel-definition-vorteile-und-besonderheiten>

Saake, G., Sattler, K., Heuer, A. (2018). Datenbanken: Konzepte und Sprachen. Germany: MITP.

SQL Abfrage: Die 5 wichtigsten Befehle, die du kennen solltest (2023). Abgerufen am 11. März, 2024, <https://datascientest.com/de/sql-abfrage-die-5-wichtigsten-befehle-die-du-kennen-solltest>

Stickel, E. (2013). Konzeptuelle Datenmodellierung. Germany: Vieweg+Teubner Verlag.

Studer, T. (2019). Relationale Datenbanken: Von den theoretischen Grundlagen zu Anwendungen mit PostgreSQL. Germany: Springer Berlin Heidelberg.

Typen von Integritätsbedingungen (2021). Abgerufen am 26. März, 2024, <https://www.ibm.com/docs/de/db2/10.5?topic=constraints-types>

Was sind Primärschlüssel und Fremdschlüssel (2023). Abgerufen am 26. März, 2024, <https://databasecamp.de/daten/primaerschluessel>

Zhao, A. (2022). SQL – kurz & gut. Germany: dpunkt.verlag.

